An Introduction to the Syntax and Content of Cyc

Cynthia Matuszek, John Cabral, Michael Witbrock, John DeOliveira

Cycorp, Incorporated
3721 Executive Center Drive
Austin, Texas 78731-1615
{cynthia, jcabral, witbrock, johnd}@cyc.com

Abstract

From the beginning, a primary goal of the Cyc project has been to build a large knowledge base containing a store of formalized background knowledge suitable for supporting reasoning in a variety of domains. In this paper, we will discuss the portion of Cyc technology that has been released in open source form as OpenCyc, provide examples of the content available in ResearchCyc, and discuss their utility for the future development of fully formalized knowledge bases.

Introduction

The Cyc Project and Knowledge Representation

From the beginning, a primary goal of the Cyc project has been to build a large knowledge base containing a store of formalized background knowledge suitable for a variety of reasoning and problem-solving tasks in a variety of domains. The thesis underlying Cyc is that, while systems containing only special-purpose domain knowledge have accomplished extraordinary things in a variety of fields, these systems can be brittle [Friedland et al 2004] and difficult to extend to new or unforeseen problems or problem areas [Lenat 1995]. This is particularly relevant in any area involving natural-language interactions or question answering [Hovy et al 2002], where the breadth of the problem space is often difficult or impossible to fully define beforehand.

The Cyc project has spent the past twenty years—approximately 900 person-years of effort—building a knowledge base that is intended to capture a broad selection of common-sense background knowledge. This knowledge base, or "KB," is intended to support unforeseen (and even unforeseeable) future knowledge representation and reasoning tasks. In that time, various applications—including CycSecure [Shepard et al 2005], a very early Thesaurus Manager, and Cyc's Digital Aristotle application [Friedland et al 2004]—have been built using ontologies or slices of ontology from the Cyc KB.

In this paper, we will describe the language CycL, present an overview of the Cyc ontology that is expressed in that language, and discuss how this reflects the past twenty years of learning experiences in knowledge representation. Changes in the ontological structure and content have come about partly as a result of developments in our understanding of the role of background knowledge, but also partly because the existence of *some* background knowledge fundamentally changes the mechanisms by which more knowledge can be added.

Cyc is a mature, but still developing, technology. The Cyc KB contains more than 2.2 million assertions (facts and rules) describing more than 250,000 terms, including nearly 15,000 predicates. OpenCyc is a freely available subset of the knowledge base. It also includes a freely available, executable Knowledge Server that includes an inference engine and other tools for accessing, utilizing, and extending the content of the knowledge base. Downloads, documentation, and other materials are available at http://www.opencyc.org.

In its current version, OpenCyc includes an ontology of 47,000 terms, which are defined and elaborated using 306,000 assertions. Although this is a small subset of the total content of Cyc, OpenCyc provides two potentially valuable resources for those interested in developing ontologies and knowledge bases. It provides a foundation of concepts that can be immediately used and easily extended; and it provides CycL, an expressive language that supports the OpenCyc ontology. As a side effect, the definitional vocabulary used to express the taxonomic information in OpenCyc reflects many of the expression needs and issues encountered over the course of building the Cyc KB.

The CycL Language

CycL's syntax expresses a number of extensions to first order-logic [Ramachandran et al 2005]. It includes a quoting mechanism to enable differentiation between knowledge involving a concept, as opposed to knowledge about the term that expresses the concept. So, while the ontology can include the knowledge that all dogs are mammals, it is also possible to represent facts about the *term* used to denote that species (i.e., `#$Dog`), such as when it was created and by whom. CycL also has a number of features of higher-order logics. Most notably, there is quantification over predicates, functions, and sentences, and there is the ability for predicates to take predicates as values, including themselves in some cases.

Quoting and higher-order extensions are an important aspect of CycL's utility for the development of formalized knowledge bases. They allow the representation of the semantics of the language's terms to be represented within the same language as all other knowledge, so that all knowledge has a common representational basis and is mutually accessible to inference. The increases in expressivity reflect practical experience with knowledge representation and the needs of ontological engineers. In particular, these higher order features allow for such things as the introduction of 'rule macro predicates', which allow for compact expressions of relationships that would otherwise require rules. The next sections will discuss the representation of the semantics of terms within the ontology. This will be followed by some examples of the content of the OpenCyc ontology. Finally, the relationship between OpenCyc and ResearchCyc will be described.

Examples of Semantic Constraint Vocabulary

OpenCyc is, at heart, a taxonomic projection of the Cyc Knowledge Base—it primarily describes Cyc's taxonomic and definitional ontology. *Taxonomic* knowledge is the knowledge of class membership of terms, and the subsumption relationships among those classes (expressed with the predicates isa and genls, respectively). *Definitional* knowledge is primarily knowledge of the intended meaning of predicate and function vocabulary. There is a subsumption relationship among predicates, where the holding of one relation to some set of arguments implies the holding of another relationship to those arguments (expressed via genlPreds).

The vast majority of the definitional knowledge about predicates, however, pertains to semantic correctness and applicability. If a CycL sentence conforms to the grammar but not its semantic requirements, it is said to be semantically ill-formed.[1] Semantic well-formedness implements constraints on the "common sense" use of relationships, and helps automate the validation of knowledge, by rejecting the formal equivalents of ridiculous sentences.

The most basic type of definitional information is the argument constraint: knowledge of what types of entities can be the values of a predicate or function. For instance, the predicate biologicalMother is a binary predicate that relates an animal to the female animal that gave birth to it. It has these definition assertions:

```
(isa biologicalMother IrreflexiveBinaryPredicate)
(arg1Isa biologicalMother Animal)
(arg2Isa biologicalMother FemaleAnimal)
```

[1] As a simple example, consider the English sentence "Julia gave birth to a prime number." It is grammatically well-formed but it expresses a confused idea that could not be true in any context.

In English, the first sentence states that nothing can bear the relation biologicalMother to itself. That sentence reflects an important representational strategy in the Cyc ontology; while that feature of a relation could be represented with a logical axiom, the pattern of such an axiom is uselessly repetitive. Therefore, we can avoid the necessity of introducing many of those axioms by introducing the collection Irreflexive-BinaryPredicate, expressing that logical feature in an axiom that references the collection, and making new predicates instances of the collection.

The second and third sentences describing biologicalMother assert that any sentence using that predicate take an animal as its first value and a female animal as its second value. The definitional axiom for the predicate arg1isa is expressed in CycL as:

```
(forAll ?PRED (forAll ?TYPE
    (forAll ?X (forAll ?VARS
      (implies
        (and
          (arg1Isa ?PRED ?TYPE)
          (?PRED ?X ?VARS))
        (isa ?X ?TYPE))))))
```

Predicates like arg1Isa and arg2Isa work in conjunction with the taxonomic knowledge in OpenCyc to define proper usage of the vocabulary.

Similarly, taxonomic knowledge is characterized by the predicates genls and disjointWith. The predicate genls expresses inheritance among types and structures the ontology, while disjointWith states that two collections do not share any members. The predicate isa is another primitive relation, which expresses membership in classes (CycL collections). Its definitional assertions include:

```
(isa isa BinaryPredicate)
(arg1Isa isa Thing)
(arg2Isa isa Collection)
```

The predicate genls is the basic sub-class predicate and it is the basis of the taxonomic hierarchy. Its definitional assertions are:

```
(isa genls TransitiveBinaryPredicate)
(isa genls ReflexiveBinaryPredicate)
(arg1Isa genls Collection)
(arg2Isa genls Collection)
(arg1Genl genls Thing)
(arg2Genl genls Thing)
```

The axiom that defines genls is:

```
(forAll ?TYPE1 (forAll ?TYPE2 (forAll ?X
  (implies
    (and
      (genls ?TYPE1 ?TYPE2)
      (isa ?X ?TYPE1))
    (isa ?X ?TYPE2)))))
```

In addition to these predicates, which help define concepts in terms of their relation to other CycL terms, a large amount of knowledge in OpenCyc is stored in the predicate comment, the main documentation predicate. Its definitional assertions have an interesting feature that reflect using quoted knowledge to support knowledge management:

```
(arg1QuotedIsa comment CycLIndexedTerm)
(arg2QuotedIsa comment SubLString)
```

These semantic well-formedness assertions invoke quotation, so the class restrictions on the arguments are not relative to the concepts denoted by the terms, but to the terms themselves. Therefore, knowledge for managing the ontology (e.g., documentation) is properly represented within the ontology as well.

A common criticism of OpenCyc is that although it includes a large number of predicates, most of those predicates are not fully ‘populated’, nor are rules included that axiomatize the meaning of the terms. Although there is merit to those criticisms, they fail to acknowledge that a huge body of such knowledge is encoded in the definitional assertions that form the basis for OpenCyc. Taxonomic and definitional knowledge restrict the possible usage of the vocabulary and, as a result, help users share knowledge while maintaining semantic well-formedness in the knowledge exchanged. Those constraints are more efficient than if they were to be ubiquitously axiomatized.

Background knowledge (for example, that it is ridiculous for a person to give birth to a number) can be represented in a number of ways. For instance, a rule to that effect could be written. However, a strong ontology of high-level knowledge, with inheritance, allows for a more efficient and thorough expression of common sense facts.

The terms mentioned are intended as examples of the taxonomic knowledge available in OpenCyc, which includes a large number of predicates that serve similar roles. To provide additional examples, vocabulary exists for restricting arguments to sub-classes, for stating that the result of a functional expression bears some relationship to one of the arguments, or that the arguments of a collection must share some relationship or properties.

The Content of the Cyc Ontology

Since one purpose of the Cyc ontology is to enable the usage of knowledge across domains, the ontology includes a wide range of categories. One of the fundamental distinctions in the ontology is between collections and individuals. The difference between collections and individuals plays an important role in the development of the ontology. For example, although historical knowledge is typically focused on individual events, countries, and persons, chemical and other scientific knowledge is typically described in terms of the properties of an entire class. (The predicates that are used in a domain tend to reflect whether the knowledge is typically expressed at the level of the individual or the collection.)

The Cyc *upper ontology* is aimed at regulating a number of different aspects of the ontology itself. Like other ontologies, the OpenCyc upper ontology allows for the representation of individuals and their relation to space, time, and human perception. However, the ontology also includes a large number of collections of collections, or *types*. This enables the categorization of collections based on important properties that are true of their instances. For example, the collections ExistingObjectType and ExistingStuffType are second-order collections whose instances correspond to the count or mass interpretations of nouns. In the case of the latter, we know that, to some granularity, any portion of an instance of the class is also an instance of the class.

Another feature of the OpenCyc ontology is a large number of functions that allow for the compositional expression of new concepts. For instance, the function SubcollectionOfWithRelationToFn is used to denote new collections based on existing concepts. The functional expression

```
(SubcollectionOfWithRelationToFn Ambassador
    basedInRegion Turkey)
```

is the collection of all ambassadors to Turkey.

The Upper, Middle, and Lower Cyc Ontology

The Cyc KB is traditionally subdivided into the upper, middle, and lower ontologies. These distinctions are intended to capture the level of generality of the information contained within them; the upper ontology is limited to broad, abstract, or highly structural concepts, and as such is the smallest but most broadly referenced area of the Cyc ontology. It is intended to capture concepts such as temporality, mathematics, and relationship types.[2] The middle ontology is intended to capture a layer of abstraction that is widely used, but not universal to all knowledge engineering efforts; examples might be geospatial relationships, broad knowledge of human interactions, or everyday items and events. The lower ontology contains domain-specific “leaf level” knowledge, such as that specific to a field of study like chemistry, or information about a particular person or nation. This layer accounts for the largest segment of knowledge in the KB, but the least broadly applicable.

Some examples of the existing knowledge available from the OpenCyc and ResearchCyc ontologies can help to provide an idea what kinds of information have gone into

[2] A graphical view of some selected components of the Upper Ontology, showing their connections, can be seen at: http://www.cyc.com/cycdoc/vocab/upperont-diagram.html

creating a repository of background knowledge, as well as exposing more of the CycL language and KB structure. The following are assertions drawn from the Cyc Knowledge Base; concepts are presented which have required special care, or which have required reworking in the face of additional knowledge over time. Representative assertions have been selected from each concept.

Upper Ontology Examples: Event and Situation

(comment **Event** "An important specialization of Situation, and thus also of IntangibleIndividual and TemporallyExistingThing. Each instance of Event is a dynamic situation in which the state of the world changes; each instance is something one would say 'happens'. Events are intangible because they are changes per se, not tangible objects that effect and undergo changes. Notable specializations of Event include Event-Localized, PhysicalEvent, Action, and GeneralizedTransfer. Events should not be confused with TimeIntervals.")

(isa Event TemporalStuffType)

(isa Event Collection)

(quotedIsa Event PublicConstant-CommentOK)

(quotedIsa Event VocabularyConstrainingAbstraction)

(genls Event Situation)

(disjointWith Event PositiveDimensionalThing)

(genls InstantaneousEvent Event)

(genls HelicopterLanding Event)
inferred knowledge

(genls (BecomingFn Intoxicated) Event)

(relationExistsAll victim Event Victim-UnfortunatePerson)
For every instance of the collection Victim-UnfortunatePerson, there exists an Event in which that person was the victim—i.e., an event for which these statements hold:
(victim ?SOMEVICTIM ?SOMEEVENT)
(isa ?SOMEVICTIM Victim-UnfortunatePerson)
(isa ?SOMEVICTIM Event))

(comment **Situation** "A specialization of both IntangibleIndividual and TemporalThing. Each instance of Situation is a state or event consisting of one or more objects having certain properties or bearing certain relations to each other. Notable specializations of Situation are Event and StaticSituation; it is disjoint with SomethingExisting.")

(genls Situation TemporallyExtendedThing)

(genls Situation TemporallyExistingThing)

(genls Situation IntangibleIndividual)

(disjointWith Situation SpatialThing)

```
(implies
  (and
   (isa ?AGT Agent-Generic)
   (causes-Underspecified ?AGT ?EVT)
   (isa ?EVT Situation))
  (causalActors ?EVT ?AGT))
```
If there is a situation that is caused (in some sense) by some agent, that agent plays the role of causal actor.

```
(implies
  (and
    (isa ?SIT Situation)
    (providesMotiveFor ?SIT ?AGENT
      ?EVENT-TYPE ?ROLE))
  (increases-Generic ?SIT
    (relationExistsInstance ?ROLE
      ?EVENT-TYPE ?AGENT) likelihood))
```
If some situation provides a motive for an agent to play a certain role in some kind of event, the likelihood of that event occurring increases.

Middle Ontology Example: SocialGathering

(comment SocialGathering "A specialization of SocialOccurrence. Each instance of SocialGathering is an intentional social gathering of people who have the same or similar purposes in attending, and in which there is communication between the participants. Specializations include BabyShower, Carnival, and Rally. Note that a group of people waiting to board an elevator is not typically a SocialGathering, even though they share a common purpose, since they are not expected to talk to each other.")

(disjointWith SocialGathering SingleDoerAction)

(disjointWith SocialGathering ConflictEvent)

(disjointWith SocialGathering IntrinsicStateChangeEvent)

(keStrongSuggestionPreds SocialGathering dateOfEvent)
Although it is not semantically required, it is likely that getting a dateOfEvent assertion for any given instance of SocialGathering would be appropriate or desirable.

(requiredActorSlots SocialGathering attendees)
In every social occasion something must play the role of attendees.

Lower Ontology Example: Chemistry Terms

(comment **ChemicalReaction** "A collection of events; a subcollection of PhysicalTransformationEvent. Each instance of ChemicalReaction is an event in which two or more substances undergo a chemical change, i.e., some portions of the substances involved are transformed into different ChemicalSubstanceTypes. The transformations are brought about by purely chemical (including biochemical) means which affect chemical bonds between atoms in the molecules of stuff. Examples of ChemicalReaction: instances of CombustionProcess; instances of Photosynthesis-Generic.")

(keGenlsStrongSuggestionPreds-RelationAllExists ChemicalReaction catalyst)

(genls ChemicalReaction PhysicalTransformationEvent)

(genls CombustionReaction ChemicalReaction)

(genls RNASplicingProcess ChemicalReaction)

(genls ExothermicReaction ChemicalReaction)

(genls ChemicalSynthesis ChemicalReaction)

```
(genls ChemicalBonding ChemicalReaction)

(genls Fermenting ChemicalReaction)

(arg1Genl availableReactantTypeInReactionType
    ChemicalReaction)
```

The first argument to an assertion made using the predicate availableReactantTypeInReactionType must generalize to some chemical reaction.

```
(comment CombustionReaction "A specialization of both
    ChemicalReaction and CombustionProcess. Each
    instance of CombustionReaction is a rapid chemical
    reaction that produces a flame. Many combustion
    reactions involve oxygen from the air as a reactant.")

(genls CombustionReaction CombustionProcess)

(genls CombustionReaction ChemicalReaction)

(outputsCreated-TypeType CombustionReaction Flame)
```

Events which are CombustionReactions have members of the collection Flame as outputs.

```
(comment catalyst "The predicate catalyst identifies the
    particular thing that acts as a catalyst in a particular
    chemical reaction. (catalyst R X) means that the
    ChemicalReaction R has the particular quantity of
    substance X as a catalyst. For example, every instance
    of Photosynthesis-Generic has some portion of
    Chlorophyll as a catalyst; an amount of Water may be a
    catalyst in some OxidationProcess of a Metal.")

(isa catalyst ActorSlot)

(genlPreds catalyst unchangedActors)

(arg1Isa catalyst ChemicalReaction)

(arg2Isa catalyst PartiallyTangible)

(interArgIsa2-1 catalyst Chlorophyll
    PhotochemicalEnergyTransduction)
```

Sentences in which catalyst is the predicate and chlorophyll is the first argument must have some photochemical energy process as the second argument.

```
((TypeCapableFn behaviorCapable) Enzyme
    ChemicalReaction catalyst))
```

Enzymes are capable of being catalysts in chemical reactions.

An Example Application of CycL

The true test of the significance of repository of knowledge is whether it can be used to describe a novel event that includes familiar, commonsense concepts and people. As an example, the following is a partial representation of the historical event of the burning of a papal document by Martin Luther in Wittenburg. New terms are underlined when they are first introduced; all other terms are pre-existing:

```
(isa BurningOfPapalBull SocialGathering)
```

Because this is an instance of SocialGathering, it is an known to be an instance of Event and to have attendees.

```
(eventOccursAt BurningOfPapalBull
    CityOfWittenburgGermany)

(dateOfEvent BurningOfPapalBull
    (DayFn 10 (MonthFn December (YearFn 1520))))

(attendee BurningOfPapalBull
    MartinLuther-ReligiousFigure)
```

Martin Luther is already represented in the KB, along with basic biographical information such as birth and death date, country of residence, and native language.

```
(relationInstanceExistsMin BurningOfPapalBull
    attendees UniversityStudent 40)
```

At least forty university students attended the event. RelationInstanceExistsMin is a rule macro predicate.

```
(isa BurningOfPapalBull-Document CombustionProcess)

(properSubEvent BurningOfPapalBull-Document
    BurningOfPapalBull)

(relationInstanceExists
    inputsDestroyed BurningOfPapalBull-Document
    (CopyOfConceptualWorkFn
        PapalBull-ExcommunicationOfLutherCW))
```

The thing destroyed is a member of the functionally defined collection "all copies of the conceptual work PapalBull-ExcommunicationOfLuther". The distinction between the conceptual *artifact and the* specific copy *being burned prevents Cyc from concluding that the conceptual work has been utterly destroyed (in the same way that burning a copy of Moby Dick does not destroy the work Moby Dick generally).*

```
(thereExists ?EVT
    (and
        (performedBy ?EVT
            MartinLuther-ReligiousFigure)
        (causes ?EVT BurningOfPapalBull-Document)))
```

The actual burning of a document was caused by some additional sub-event EVT (such as holding a match to it or throwing it into a fire). Unlike the distinction between the social gathering and the actual burning of the document, both of which are reified events, this event is defined implicitly, because there is nothing else to say about it.

While there is much more to say about this event—such as its historical relevance and symbolic meaning[3], the definition of the conceptual work, and the historical framework of the event—the example gives some idea how the structural ontology present in the Cyc KB can be leveraged effectively in novel representations.

OpenCyc and ResearchCyc

Obtaining and representing sufficient background knowledge to allow efficient representation of problem-specific domain knowledge is an issue facing researchers working in essentially all knowledge-based fields, including KR&R, question answering, and knowledge management. The lack of a centralized source of accumulated knowledge both increases the effort necessary for any such research project, and makes sharing knowledge among groups more difficult [Gruber 1991].

[3] (See, e.g., symbolizes, RejectingSomething, and RomanCatholicChurch)

OpenCyc is a freely available section of some of the general knowledge in the Cyc system.[4]

The knowledge in OpenCyc is a small subset of the knowledge in the ResearchCyc KB. However, since the bulk of the knowledge is definitional, that subset represents a large body of background knowledge and provides the key mechanism that enables formal knowledge to be shared among users of the ontology. OpenCyc has been used to support research in areas ranging from automatic pruning of irrelevant knowledge [Conesa & Olivé 2004], to the integration of Semantic Web metadata [Sicilia et al 2004], to obtaining part of speech information from a body of natural language data using machine learning [O'Hara et al 2003, Matuszek et al 2005].

The OpenCyc system has been criticized for being restrictive, particularly with respect to instance-level knowledge, for lacking any reasoning capability, and being formally inconsistent. While the last is difficult to overcome in a very large, largely hand-tooled knowledge base, concerns about inadequate coverage of some types of knowledge have been at least partially addressed by the release of ResearchCyc, which is made available without charge for research use.[5]

Conclusions

In the course of developing a large common-sense knowledge base, we have developed both an extensive taxonomy of concepts and terms, and a large, self-reflective vocabulary for describing the most common definitional needs we have encountered over time. In ResearchCyc, we have also entered information describing a large range of lower-level topics and concepts that have the potential to be relevant as a source of domain and common-sense knowledge in a variety of fields. We invite other researchers to make use of the ResearchCyc and OpenCyc platforms to drive knowledge-based efforts in any circumstance where they may prove useful.

References

Conesa, J. & Olivé, A., "A General Method for Pruning OWL Ontologies", in *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2004, Agia Napa, Cyprus, October 25-29, 2004, Proceedings, Part II*. Springer-Verlag, 2004.

Friedland, N.S, Allen, P.G., Witbrock, M., Angele, J., Staab, S., Israel, D., Chaudhri, V., Porter, B., Barker, K., Clark, P., "Towards a Quantitative, Platform-Independent Analysis of Knowledge Systems," in *Proc. of the 9th International Conference on the Principles of Knowledge Representation and Reasoning*. Whistler, 507-515, 2004.

Gruber, T., 1991, "The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases," in *Principles of Knowledge Representation and Reasoning: Proc. of the 2nd International Conference*. San Mateo, CA.

Hovy, E., Hermjakob, U., Lin, C-Y., Ravichandran, D. "Using knowledge to facilitate factoid answer pinpointing Full text." In *Proc. of the 19th International Conference on Computational Linguistics, Vol. 1,* Taipei, Taiwan, 2002.

Lenat, D. "Cyc: A Large-Scale Investment in Knowledge Infrastructure, *CACM* 38, no. 11. 1995.

Matuszek, C., Witbrock, M., Kahlert, R.C., Cabral, J., Schneider, D., Shah, P., and Lenat, D. "Searching for Common Sense: Populating Cyc from the Web." In *Proc. of the 20th National Conference on Artificial Intelligence,* Pittsburgh, PA, July 2005.

O'Hara, T., Salay, N, Witbrock, M., Schneider, D., Aldag, B., Bertolo, S., Panton, K., Lehmann, F., Curtis, J., Smith, M., Baxter, D., Wagner, P. "Inducing criteria for mass noun lexical mappings using the Cyc Knowledge Base and its Extension to WordNet." In *Proc. of the 5th International Workshop on Computational Semantics,* Tilburg, 2003.

Ramachandran, D., Reagan, P., Goolsbey, K. "First-Orderized ResearchCyc: Expressivity and Efficiency in a Common-Sense Ontology." In *Papers from the AAAI Workshop on Contexts and Ontologies: Theory, Practice and Applications*. Pittsburgh, PA, July 2005.

Shepard, B., Matuszek, C., Fraser, C.B., Wechtenhiser, W., Crabbe, D., Güngördü, Z., Jantos, J., Hughes, T., Lefkowitz, L., Witbrock, M., Lenat, D., Larson, E. "A Knowledge-Based Approach to Network Security: Applying Cyc in the Domain of Network Risk Assessment." In *Proc. of the 17th Innovative Applications of Artificial Intelligence Conference,* Pittsburgh, PA, July 2005.

Sicilia, M.A., Garcia, E., Sanchez, S., Rodriguez, E. "On Integrating Learning Object Metadata inside the OpenCyc Knowledge Base," *icalt*, pp. 900-901. Fourth IEEE International Conference on Advanced Learning Technologies, 2004.

[4] ResearchCyc, which includes not only taxonomic and definitional knowledge, but also extensive instance-level population, is also available under a free research-purposes license.

[5] ResearchCyc can be obtained from http://research.cyc.com, or by contacting the authors of this paper.

The material is based upon work funded in whole or in part by the U.S. Government and any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Government.